

Refine Search

Search Results -

Terms	Documents
5937196.pn.	1

Database:

US Pre-Grant Publication Full-Text Database
 US Patents Full-Text Database
 US OCR Full-Text Database
 EPO Abstracts Database
 JPO Abstracts Database
 Derwent World Patents Index
 IBM Technical Disclosure Bulletins

Search:

L24

Search History

 DATE: Thursday, October 21, 2004 [Printable Copy](#) [Create Case](#)

<u>Set Name</u>	<u>Query</u>	<u>Hit Count</u>	<u>Set Name result set</u>
<i>DB=USPT; PLUR=YES; OP=ADJ</i>			
<u>L24</u>	5937196.pn.	1	<u>L24</u>
<u>L23</u>	5146594.pn.	1	<u>L23</u>
<u>L22</u>	17 and ((cop\$ or duplicat\$) near6 entr\$)	26	<u>L22</u>
<u>L21</u>	17 and (cop\$ near6 entr\$)	10	<u>L21</u>
<u>L20</u>	L19 and (iterat\$ near9 dataflow\$)	3	<u>L20</u>
<u>L19</u>	17 and (modif\$ or chang\$ or alter\$) near4 block\$	110	<u>L19</u>
<i>DB=TDBD; PLUR=YES; OP=ADJ</i>			
<u>L18</u>	entr\$ and exit\$ and dataflow and (modif\$ or chang\$ or alter) and (cop\$ or image\$)and iter\$ and (modif\$ or chang\$ or alter) near5 entr\$ and cop\$ near9 (entr\$)	0	<u>L18</u>
<i>DB=DWPI; PLUR=YES; OP=ADJ</i>			
<u>L17</u>	entr\$ and exit\$ and dataflow and (modif\$ or chang\$ or alter) and (cop\$ or image\$)and iter\$ and (modif\$ or chang\$ or alter) near5 entr\$ and cop\$ near9	0	<u>L17</u>

(entr\$)		
<i>DB=JPAB; PLUR=YES; OP=ADJ</i>		
<u>L16</u>	entr\$ and exit\$ and dataflow and (modif\$ or chang\$ or alter) and (cop\$ or image\$)and iter\$ and (modif\$ or chang\$ or alter) near5 entr\$ and cop\$ near9 (entr\$)	0 <u>L16</u>
<i>DB=EPAB; PLUR=YES; OP=ADJ</i>		
<u>L15</u>	entr\$ and exit\$ and dataflow and (modif\$ or chang\$ or alter) and (cop\$ or image\$)and iter\$ and (modif\$ or chang\$ or alter) near5 entr\$ and cop\$ near9 (entr\$)	0 <u>L15</u>
<i>DB=USOC; PLUR=YES; OP=ADJ</i>		
<u>L14</u>	entr\$ and exit\$ and dataflow and (modif\$ or chang\$ or alter) and (cop\$ or image\$)and iter\$ and (modif\$ or chang\$ or alter) near5 entr\$ and cop\$ near9 (entr\$)	0 <u>L14</u>
<i>DB=PGPB; PLUR=YES; OP=ADJ</i>		
<u>L13</u>	6560774.pn.	0 <u>L13</u>
<u>L12</u>	entr\$ and exit\$ and dataflow and (modif\$ or chang\$ or alter) and (cop\$ or image\$)and iter\$ and (modif\$ or chang\$ or alter) near5 entr\$ and cop\$ near9 (entr\$)	1 <u>L12</u>
<i>DB=USPT; PLUR=YES; OP=ADJ</i>		
<u>L11</u>	L10 and cop\$ near9 (entr\$)	1 <u>L11</u>
<u>L10</u>	17 and (modif\$ or chang\$ or alter) near5 entr\$	25 <u>L10</u>
<u>L9</u>	17 and (modif\$ or chang\$ or alter) near5 entr\$	0 <u>L9</u>
<u>L8</u>	L7 and l6	14 <u>L8</u>
<u>L7</u>	entr\$ and exit\$ and dataflow and (modif\$ or chang\$ or alter) and (cop\$ or image\$)and iter\$	152 <u>L7</u>
<u>L6</u>	717/131,132,140,155,156,151.ccls.	740 <u>L6</u>
<u>L5</u>	l2 and optimiz\$ and (cop\$ or image\$)	2 <u>L5</u>
<u>L4</u>	L3 and cop\$	1 <u>L4</u>
<u>L3</u>	L2 and iterat\$	4 <u>L3</u>
<u>L2</u>	dataflow\$ and entr\$ near5 (basic block\$) and exit\$ near4 (basic block\$)	10 <u>L2</u>
<u>L1</u>	6117185.pn.	1 <u>L1</u>

END OF SEARCH HISTORY



[Subscribe](#) (Full Service) [Register](#) (Limited Service, Free) [Login](#)

Search: ☒ The ACM Digital Library ☐ The Guide

SEARCH

THE ACM DIGITAL LIBRARY



[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

Terms used

dataflow and **analysis** and **copy** and **entry** and **exit** and **basic block** and **iteration**

Found **59,093** of **143,484**

Sort results
by



[Save results to a Binder](#)

[Try an Advanced Search](#)

Display
results



[Search Tips](#)

[Try this search in The ACM Guide](#)

☐ Open results in a new
window

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐

1 [A generalized theory of bit vector data flow analysis](#)

Uday P. Khedker, Dhananjay M. Dhamdhare

September 1994 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,

Volume 16 Issue 5

Full text available: pdf(2.42 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

The classical theory of data flow analysis, which has its roots in unidirectional flows, is inadequate to characterize bidirectional data flow problems. We present a generalized theory of bit vector data flow analysis which explains the known results in unidirectional and bidirectional data flows and provides a deeper insight into the process of data flow analysis. Based on the theory, we develop a worklist-based generic algorithm which is uniformly applicable to unidirectional and bidirect ...

Keywords: bidirectional data flows, data flow analysis, data flow frameworks

2 [Compiler transformations for high-performance computing](#)

David F. Bacon, Susan L. Graham, Oliver J. Sharp

December 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 4

Full text available: pdf(6.32 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

In the last three decades a large number of compiler transformations for optimizing programs have been implemented. Most optimizations for uniprocessors reduce the number of instructions executed by the program using transformations based on the analysis of scalar quantities and data-flow techniques. In contrast, optimizations for high-performance superscalar, vector, and parallel processors maximize parallelism and memory locality with transformations that rely on tracking the properties o ...


Keywords: compilation, dependence analysis, locality, multiprocessors, optimization, parallelism, superscalar processors, vectorization

3 [Efficient computation of interprocedural definition-use chains](#)

Mary Jean Harrold, Mary Lou Soffa

March 1994 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,

Volume 16 Issue 2

Full text available:  [pdf\(2.00 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

The dependencies that exist among definitions and uses of variables in a program are required by many language-processing tools. This paper considers the computation of definition-use and use-definition chains that extend across procedure boundaries at call and return sites. Intraprocedural definition and use information is abstracted for each procedure and is used to construct an interprocedural flow graph. This intraprocedural data-flow information is then propagated throughout the program ...

Keywords: dataflow testing, interprocedural dataflow analysis, interprocedural definition-use chains, interprocedural reachable uses, interprocedural reaching definitions

4 [The program dependence graph and its use in optimization](#)

Jeanne Ferrante, Karl J. Ottenstein, Joe D. Warren

July 1987 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 9 Issue 3

Full text available:  [pdf\(2.51 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

In this paper we present an intermediate program representation, called the program dependence graph (PDG), that makes explicit both the data and control dependences for each operation in a program. Data dependences have been used to represent only the relevant data flow relationships of a program. Control dependences are introduced to analogously represent only the essential control flow relationships of a program. Control dependences are derived from the ...

5 [Using dataflow analysis techniques to reduce ownership overhead in cache coherence protocols](#)

Jonas Skeppstedt, Per Stenström

November 1996 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 18 Issue 6

Full text available:  [pdf\(284.68 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#), [review](#)

In this article, we explore the potential of classical dataflow analysis techniques in removing overhead in write-invalidate cache coherence protocols for shared-memory multiprocessors. We construct the compiler algorithms with varying degree of sophistication that detect loads followed by stores to the same address. Such loads are marked and constitute a hint to the cache to obtain an exclusive copy of the block so that the subsequent store does not introduce access penalties. The simplest ...

Keywords: cache coherence, dataflow analysis, performance evaluation

6 [Global communication analysis and optimization](#)

Soumen Chakrabarti, Manish Gupta, Jong-Deok Choi

May 1996 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1996 conference on Programming language design and implementation**, Volume 31 Issue 5

Full text available:  [pdf\(1.39 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Reducing communication cost is crucial to achieving good performance on scalable parallel machines. This paper presents a new compiler algorithm for global analysis and optimization of communication in data-parallel programs. Our algorithm is distinct from existing approaches in that rather than handling loop-nests and array references one by one, it considers all communication in a procedure and their interactions under different placements

before making a final decision on the placement of any ...

7 An interval-based approach to exhaustive and incremental interprocedural data-flow analysis

Michael Burke

July 1990 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,
Volume 12 Issue 3

Full text available:  [pdf\(4.43 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

We reformulate interval analysis so that it can be applied to any monotone data-flow problem, including the nonfast problems of flow-insensitive interprocedural analysis. We then develop an incremental interval analysis technique that can be applied to the same class of problems. When applied to flow-insensitive interprocedural data-flow problems, the resulting algorithms are simple, practical, and efficient. With a single update, the incremental algorithm can accommodate any sequence of pr ...

8 Optimal code motion: theory and practice

Jens Knoop, Oliver Rüthing, Bernhard Steffen

July 1994 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,
Volume 16 Issue 4

Full text available:  [pdf\(2.02 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

An implementation-oriented algorithm for lazy code motion is presented that minimizes the number of computations in programs while suppressing any unnecessary code motion in order to avoid superfluous register pressure. In particular, this variant of the original algorithm for lazy code motion works on flowgraphs whose nodes are basic blocks rather than single statements, since this format is standard in optimizing compilers. The theoretical foundations of the modified algo ...


Keywords: t-refined flow graphs, code motion, computational optimality, critical edges, data flow analysis, elimination of partial redundancies, lifetime optimality, lifetimes of registers, nondeterministic flowgraphs

9 Symbolic array dataflow analysis for array privatization and program parallelization

Junjie Gu, Zhiyuan Li, Gyungho Lee

December 1995 **Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM)**

Full text available:  [pdf\(377.48 KB\)](#)

 [html\(2.76 KB\)](#)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

10 Debugging optimized code without being misled

Max Copperman

May 1994 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,
Volume 16 Issue 3

Full text available:  [pdf\(2.57 MB\)](#)


Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Correct optimization can change the behavior of an incorrect program; therefore at times it is necessary to debug optimized code. However, optimizing compilers produce code that impedes source-level debugging. Optimization can cause an inconsistency between where the user expects a breakpoint to be located and the breakpoint's actual location. This article describes a mapping between statements and breakpoint locations that ameliorates this problem. The mapping enables debugger b ...

11 Precise interprocedural dataflow analysis via graph reachability

Thomas Reps, Susan Horwitz, Mooly Sagiv

January 1995 **Proceedings of the 22nd ACM SIGPLAN-SIGACT symposium on Principles of programming languages**

Full text available:  [pdf\(1.51 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The paper shows how a large class of interprocedural dataflow-analysis problems can be solved precisely in polynomial time by transforming them into a special kind of graph-reachability problem. The only restrictions are that the set of dataflow facts must be a finite set, and that the dataflow functions must distribute over the confluence operator (either union or intersection). This class of probable problems includes—but is not limited to—the classical separable problems (als ...

12 Effectiveness of a machine-level, global optimizer

Mark S. Johnson, Terrence C. Miller

July 1986 **ACM SIGPLAN Notices , Proceedings of the 1986 SIGPLAN symposium on Compiler construction**, Volume 21 Issue 7

Full text available:  [pdf\(853.18 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present an overview of the design of a machine-code-level, global (intraprocedural) optimizer that supports several front-ends producing code for the Hewlett-Packard Precision Architecture family of machines. The basic optimization strategy is described, including information about the division of responsibilities between various components of the compiler. Optimization algorithms are described, including a discussion of the dataflow information they require. Measurements showing the col ...

13 Optimizing array bound checks using flow analysis

Rajiv Gupta

March 1993 **ACM Letters on Programming Languages and Systems (LOPLAS)**, Volume 2 Issue 1-4

Full text available:  [pdf\(1.02 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)


Bound checks are introduced in programs for the run-time detection of array bound violations. Compile-time optimizations are employed to reduce the execution-time overhead due to bound checks. The optimizations reduce the program execution time through elimination of checks and propagation of checks out of loops. An execution of the optimized program terminates with an array bound violation if and only if the same outcome would have resulted during the exec ...

Keywords: available checks, check hoisting, dataflow analysis, very busy checks

14 Query evaluation techniques for large databases

Goetz Graefe

June 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 2

Full text available:  [pdf\(9.37 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)


Database management systems will continue to manage large data volumes. Thus, efficient algorithms for accessing and manipulating large sets and sequences will be required to provide acceptable performance. The advent of object-oriented and extensible database systems will not solve this problem. On the contrary, modern data models exacerbate the problem: In order to manipulate large sets of complex objects as efficiently as today's database systems manipulate simple records, query-processi ...

Keywords: complex query evaluation plans, dynamic query evaluation plans, extensible database systems, iterators, object-oriented database systems, operator model of parallelization, parallel algorithms, relational database systems, set-matching algorithms, sort-hash duality

15 The program structure tree: computing control regions in linear time

Richard Johnson, David Pearson, Keshav Pingali

June 1994 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1994 conference on Programming language design and implementation**, Volume 29 Issue 6


Full text available:  [pdf\(1.68 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

In this paper, we describe the program structure tree (PST), a hierarchical representation of program structure based on single entry single exit (SESE) regions of the control flow graph. We give a linear-time algorithm for finding SESE regions and for building the PST of arbitrary control flow graphs (including irreducible ones). Next, we establish a connection between SESE regions and control dependence equivalence classes, and show how to use the algorithm to find control regions in line ...

16 Interprocedural partial redundancy elimination and its application to distributed memory compilation

Gagan Agrawal, Joel Saltz, Raja Das

June 1995 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1995 conference on Programming language design and implementation**, Volume 30 Issue 6

Full text available:  [pdf\(1.29 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Partial Redundancy Elimination (PRE) is a general scheme for suppressing partial redundancies which encompasses traditional optimizations like loop invariant code motion and redundant code elimination. In this paper we address the problem of performing this optimization interprocedurally. We use interprocedural partial redundancy elimination for placement of communication and communication preprocessing statements while compiling for distributed memory parallel machines.

17 The benefits and costs of DyC's run-time optimizations

Brian Grant, Markus Mock, Matthai Philipose, Craig Chambers, Susan J. Eggers

September 2000 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 22 Issue 5

Full text available:  [pdf\(1.59 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

DyC selectively dynamically compiles programs during their execution, utilizing the run-time-computed values of variables and data structures to apply optimizations that are based on partial evaluation. The dynamic optimizations are preplanned at static compile time in order to reduce their run-time cost; we call this staging. DyC's staged optimizations include (1) an advanced binding-time analysis that supports polyvariant specialization (enabling both single-way and multi ...

Keywords: dynamic compilation, specialization

18 Session S6.2: compilers and program analysis: Scenario-based software characterization as a contingency to traditional program profiling

Jeffrey T. Russell, Margarida F. Jacome

October 2002 **Proceedings of the 2002 international conference on Compilers,**

architecture, and synthesis for embedded systems

Full text available:  pdf(142.05 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Program profiling is common way to characterize program behavior based on representative input. Some software, especially in embedded systems, cannot be profiled do to lack of tools or problems introduced by instrumentation of the code. As an alternative to traditionally profiling, a static analysis technique is proposed that allows a designer to characterize the flow of control of software. Operating on a flow graph representation of software, the proposed technique assists an expert designer in ...

Keywords: constraint, control flow, embedded system, performance, predicate, profiling, program profile, scenario, static analysis, typical behavior

19 [A balanced code placement framework](#)

Reinhard von Hanxleden, Ken Kennedy

September 2000 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 22 Issue 5

Full text available:  pdf(524.13 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Give-N-Take is a code placement framework which uses a generic producer-consumer mechanism. An instance of this could be a communication step between a processor that computes (produces) some data, and other processors that subsequently reference (consume) these data in an expression. An advantage of Give-N-Take over traditional partial redundancy elimination techniques is its concept of production regions, instead of single locations, which can be beneficial for general la ...

Keywords: Fortran D, Tarjan intervals, data-flow analysis, high performance Fortran, latency hiding, partial redundancy elimination

20 [An efficient ILP-based scheduling algorithm for control-dominated VHDL descriptions](#)

Michael Münch, Norbert Wehn, Manfred Glesner

October 1997 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 2 Issue 4

Full text available:  pdf(375.99 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

To adopt behavioral synthesis techniques in existing design flows, the synthesis methodology must provide the designer with a mechanism to specify a component's interface timing. This will permit pre- and postsynthesis validation through cosimulation with other subsystems or even through formal verification. In control-flow dominated designs, additional timing constraints will result in a complex specification/constraint system for which the scheduling problem has been shown to be NP-comple ...

Keywords: integer linear programming (ILP), scheduling, timing constraints

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

THE ACM DIGITAL LIBRARY


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

Terms used

iterative and dataflow and entry and exit and copy and basic block

Found 39,284 of 143,484

Sort results by


[Save results to a Binder](#)
[Try an Advanced Search](#)
[Try this search in The ACM Guide](#)

Display results


[Search Tips](#)
☐ Open results in a new window

 Results 161 - 180 of 200 Result page: [previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

 Relevance scale ☐ ☐ ☐ ☐ ☐

161 [Post-compaction register assignment in a retargetable compiler](#)

Philip Sweany, Steven Beaty

 November 1990 **Proceedings of the 23rd annual workshop and symposium on Microprogramming and microarchitecture**

 Full text available: [pdf\(998.12 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

We discuss graph-coloring register assignment in a retargetable compiler for Long-Instruction-Word architectures. Of specific concern is when, during the compilation process, should register assignment be performed. We conclude that, for best results, register assignment should follow compaction. We discuss methods of circumventing the implementation problems inherent in such late register assignment.

162 [Portable run-time support for dynamic object-oriented parallel processing](#)

Andrew S. Grimshaw, Jon B. Weissman, W. Timothy Strayer

 May 1996 **ACM Transactions on Computer Systems (TOCS)**, Volume 14 Issue 2

 Full text available: [pdf\(2.21 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Mentat is an object-oriented parallel processing system designed to simplify the task of writing portable parallel programs for parallel machines and workstation networks. The Mentat compiler and run-time system work together to automatically manage the communication and synchronization between objects. The run-time system marshals member function arguments, schedules objects on processors, and dynamically constructs and executes large-grain data dependence graphs. In this article we present ...

Keywords: MIMD, dataflow, distributed memory, object-oriented, parallel processing

163 [Evaluation of predicated array data-flow analysis for automatic parallelization](#)

Sungdo Moon, Mary W. Hall

 May 1999 **ACM SIGPLAN Notices , Proceedings of the seventh ACM SIGPLAN symposium on Principles and practice of parallel programming**, Volume 34 Issue 8

 Full text available: [pdf\(1.54 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper presents an evaluation of a new analysis for parallelizing compilers called *predicated array data-flow analysis*. This analysis extends array data-flow analysis for

[IEEE HOME](#) | [SEARCH IEEE](#) | [SHOP](#) | [WEB ACCOUNT](#) | [CONTACT IEEE](#)[Membership](#) | [Publications/Services](#) | [Standards](#) | [Conferences](#) | [Careers/Jobs](#)**IEEE Xplore®**
RELEASE 1.8Welcome
United States Patent and Trademark Office[Help](#) | [FAQ](#) | [Terms](#) | [IEEE Peer Review](#)[Quick Links](#)**Welcome to IEEE Xplore®**

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced
- ☐ CrossRef

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

IEEE Enterprise

- ☐ Access the IEEE Enterprise File Cabinet

Your search matched **0** of **1082760** documents.A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance Descending** order.**Refine This Search:**

You may refine your search by editing the current search expression or entering a new one in the text box.

☐ Check to search within this result set**Results Key:****JNL** = Journal or Magazine **CNF** = Conference **STD** = Standard**Results:****No documents matched your query.** **Print Format**[Home](#) | [Log-out](#) | [Journals](#) | [Conference Proceedings](#) | [Standards](#) | [Search by Author](#) | [Basic Search](#) | [Advanced Search](#) | [Join IEEE](#) | [Web Account](#) | [New this week](#) | [OPAC Linking Information](#) | [Your Feedback](#) | [Technical Support](#) | [Email Alerting](#) | [No Robots Please](#) | [Release Notes](#) | [IEEE Online Publications](#) | [Help](#) | [FAQ](#) | [Terms](#) | [Back to Top](#)

Copyright © 2004 IEEE — All rights reserved

IEEE HOME | SEARCH IEEE | SHOP | WEB ACCOUNT | CONTACT IEEE


[Membership](#) | [Publications/Services](#) | [Standards](#) | [Conferences](#) | [Careers/Jobs](#)
IEEE Xplore®
 RELEASE 1.8

 Welcome
 United States Patent and Trademark Office

[Help](#) | [FAQ](#) | [Terms](#) | [IEEE Peer Review](#)
[Quick Links](#)
Welcome to IEEE Xplore®

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced
- ☐ CrossRef

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

IEEE Enterprise

- ☐ Access the IEEE Enterprise File Cabinet

 Your search matched **0** of **1082760** documents.

 A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance Descending** order.

Refine This Search:

You may refine your search by editing the current search expression or entering a new one in the text box.

☐ Check to search within this result set

Results Key:
JNL = Journal or Magazine **CNF** = Conference **STD** = Standard

Results:
No documents matched your query.
Print Format
[Home](#) | [Log-out](#) | [Journals](#) | [Conference Proceedings](#) | [Standards](#) | [Search by Author](#) | [Basic Search](#) | [Advanced Search](#) | [Join IEEE](#) | [Web Account](#) | [New this week](#) | [OPAC Linking Information](#) | [Your Feedback](#) | [Technical Support](#) | [Email Alerting](#) | [No Robots Please](#) | [Release Notes](#) | [IEEE Online Publications](#) | [Help](#) | [FAQ](#) | [Terms](#) | [Back to Top](#)

Copyright © 2004 IEEE — All rights reserved

IEEE HOME | SEARCH IEEE | SHOP | WEB ACCOUNT | CONTACT IEEE


[Membership](#) [Publications/Services](#) [Standards](#) [Conferences](#) [Careers/Jobs](#)
IEEE Xplore®
 RELEASE 1.8

 Welcome
 United States Patent and Trademark Office


> Se

[Help](#) [FAQ](#) [Terms](#) [IEEE Peer Review](#)
[Quick Links](#)

Welcome to IEEE Xplore®

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced
- ☐ CrossRef

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

IEEE Enterprise

- ☐ Access the IEEE Enterprise File Cabinet

Print Format

 Your search matched **7** of **1082760** documents.

 A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance Descending** order.

Refine This Search:

You may refine your search by editing the current search expression or entering new one in the text box.

☐ Check to search within this result set

Results Key:

JNL = Journal or Magazine **CNF** = Conference **STD** = Standard

1 Contention-conscious transaction ordering in embedded multiprocessors
Khandelia, M.; Bhattacharyya, S.S.;

Application-Specific Systems, Architectures, and Processors, 2000. Proceedings IEEE International Conference on , 10-12 July 2000

Pages:276 - 285

[\[Abstract\]](#) [\[PDF Full-Text \(188 KB\)\]](#) IEEE CNF

2 An efficient timing model for hardware implementation of multirate dataflow graphs
Chandrachoodan, N.; Bhattacharyya, S.S.; Liu, K.J.R.;

Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01). 2 IEEE International Conference on , Volume: 2 , 7-11 May 2001

Pages:1153 - 1156 vol.2

[\[Abstract\]](#) [\[PDF Full-Text \(388 KB\)\]](#) IEEE CNF

3 Conditional and iterative structures using a homogeneous static dataflow graph model
Verdoscia, L.; Vaccaro, R.;

Massively Parallel Computing Systems, 1994., Proceedings of the First International Conference on , 2-6 May 1994

Pages:391 - 401

[\[Abstract\]](#) [\[PDF Full-Text \(888 KB\)\]](#) IEEE CNF

4 RPT: a CASE environment supporting the rapid prototyping approach to software development
Rizman, K.; Rozman, I.; Verber, D.;

CompEuro '91. 'Advanced Computer Technology, Reliable Systems and